«Talento Tech»

Automation Testing

PLAN DE ESTUDIOS 2025

Buenos Aires aprende
Agencia de Habilidades para el Futuro

Buenos Aires Ciudad

MODALIDAD VIRTUAL

1 Clase virtual sincrónica semanal de 2 hs (obligatoria).
1 Clase de consulta de 1 h (opcional).

ACREDITACIÓN

- 70% de asistencia a clases virtuales semanales.
- Aprobación de cuestionarios parciales de contenidos.
- Entrega y aprobación de Proyecto Final Integrador.

DURACIÓN

1 cuatrimestre de 16 clases.

CERTIFICACIÓN

Microcredencial oficial del Ministerio de Educación de la Ciudad de Buenos Aires.

Fundamentación

En el ecosistema tecnológico actual, donde el software es un componente crítico en prácticamente todos los sectores, la calidad se ha convertido en un factor determinante para el éxito empresarial. El testing manual, aunque importante, ya no es suficiente para satisfacer las demandas de velocidad, eficiencia y cobertura que requieren los ciclos de desarrollo modernos. En este contexto, la automatización de pruebas emerge como una habilidad fundamental que permite no solo mejorar la calidad del software, sino también optimizar recursos y acelerar los tiempos de entrega.
El Automation Testing proporciona un enfoque sistemático para verificar el comportamiento del software a través de scripts automatizados, permitiendo ejecutar pruebas de manera repetitiva, consistente y a gran escala. Python, con su sintaxis clara y su amplio ecosistema de bibliotecas especializadas como Selenium, Pytest y Behave, se ha posicionado como uno de los lenguajes más adecuados para la automatización de pruebas, tanto para interfaces de usuario como para APIs y servicios.

Este curso está diseñado para proporcionar a los estudiantes las competencias necesarias para convertirse en Automation Testers efectivos, combinando fundamentos sólidos de programación en Python con técnicas especializadas de automatización de pruebas. A través de un enfoque práctico y progresivo, los participantes aprenderán a diseñar, implementar y mantener frameworks de automatización robustos que puedan integrarse en flujos de trabajo de Integración Continua y Entrega Continua (CI/CD).
En un mercado laboral donde la demanda de profesionales de QA con habilidades de automatización crece constantemente, este curso representa una oportunidad valiosa para adquirir competencias altamente cotizadas en la industria tecnológica, abriendo puertas a roles como QA Automation Engineer, SDET (Software Development Engineer in Test) o Test Automation Specialist.

Contenidos mínimos

El curso está estructurado de manera progresiva, partiendo desde los fundamentos básicos de programación hasta llegar a la implementación de frameworks completos de automatización.
Los contenidos mínimos incluyen:

- Fundamentos de Programación con Python
- Fundamentos de Testing y Automatización
- Selenium WebDriver y Técnicas de Automatización Web
- Automatización de APIs y Generación de Reportes
- Integración con Metodologías Ágiles y CI/CD

Objetivo general

El curso tiene como objetivo formar profesionales capaces de diseñar e implementar soluciones efectivas de automatización de pruebas utilizando Python y herramientas especializadas como Selenium, Pytest y Behave, contribuyendo a mejorar la calidad y eficiencia en los procesos de desarrollo de software a través de la aplicación de buenas prácticas y patrones de diseño específicos para testing.

Objetivos específicos

- Proporcionar una base sólida en programación Python orientada a la automatización de pruebas, incluyendo el manejo eficiente de control de versiones con Git.
- Desarrollar competencias en el diseño e implementación de tests automatizados para interfaces web utilizando Selenium WebDriver, aplicando patrones de diseño como Page Object Model.
- Capacitar en técnicas avanzadas de testing, incluyendo parametrización de pruebas, data-driven testing y automatización de APIs REST.
- Integrar conceptos de Behavior-Driven Development (BDD) y herramientas como Behave para facilitar la colaboración entre equipos técnicos y de negocio.
- Formar en la implementación de soluciones de Integración Continua (CI) para la ejecución automática de tests, utilizando GitHub Actions como plataforma de referencia.
- Desarrollar habilidades para la creación y mantenimiento de frameworks de automatización completos, incluyendo sistemas de reportería y gestión de errores.

Resultados de aprendizaje

Al finalizar este curso, los estudiantes serán capaces de:

Diseñar e implementar soluciones de automatización de pruebas eficientes utilizando Python como lenguaje principal. Habrán desarrollado competencias en el uso de herramientas especializadas como Selenium WebDriver para automatizar interfaces web, Pytest como framework de testing, y Behave para implementar pruebas basadas en comportamiento (BDD).

Los participantes podrán aplicar el patrón Page Object Model para crear frameworks de automatización mantenibles y escalables, implementar pruebas parametrizadas y data-driven para maximizar la cobertura, y automatizar pruebas de APIs REST utilizando la biblioteca Requests para validar servicios back-end.

Los estudiantes serán capaces de generar reportes detallados de ejecución de pruebas e implementar estrategias de logging y captura de errores para facilitar la depuración. También podrán integrar sus soluciones de automatización en pipelines de CI/CD utilizando GitHub Actions, contribuyendo a la implementación de prácticas de Integración Continua en equipos de desarrollo.

Al completar el curso, los participantes no solo dominarán las habilidades técnicas necesarias para el Automation Testing, sino que también comprenderán el rol estratégico que desempeña la automatización de pruebas en la mejora de la calidad del software y la eficiencia del proceso de desarrollo, posicionándose como profesionales valiosos en el mercado laboral actual.

»Automation Testing

Temario

CLASE 01
Introducción al Automation Testing

CLASE 02
Fundamentos de Python - Parte 1

CLASE 03
Fundamentos de Python - Parte 2 y Control de Versiones

CLASE 04
Introducción a Pytest y Automatización Básica

CLASE 05
Introducción a HTML y Estructura de Páginas Web

CLASE 06
DOM y CSS para Automatización

CLASE 07
Introducción a Selenium WebDriver

CLASE 08
Localización de Elementos y Acciones en Selenium

CLASE 09
Page Object Model

CLASE 10
Manejo de Datos de Prueba

CLASE 11
Automatización de Pruebas de API - Parte 1

CLASE 12
Automatización de Pruebas de API - Parte 2

CLASE 13
Reportes y Logging

CLASE 14
Introducción a BDD y Behave

CLASE 15
Integración Continua (CI/CD)

CLASE 16
Presentación del Proyecto Final y Cierre

Temario

CLASE 01 » Introducción al Automation Testing

- Presentación del curso: objetivos, estructura, metodología y evaluación.
- ¿Qué es un Automation Tester?
- Rol del Automation Tester en el ciclo de desarrollo.
- Habilidades requeridas.
- Mercado laboral y oportunidades.
- Instalación del entorno básico:
  - Instalación de Python
  - Instalación de VS Code
  - Configuración inicial

CLASE 02 » Fundamentos de Python - Parte 1

- Python desde cero:
  - Variables y tipos de datos
  - Operadores
  - Entrada/Salida básica
- Estructuras de control
- Ejercicios prácticos básicos

CLASE 03 » Fundamentos de Python - Parte 2 y Control de Versiones

- Estructuras de datos:
  - Listas, tuplas y diccionarios
  - Operaciones comunes
- Funciones:
  - Definición y llamada
  - Parámetros y retorno de valores
- Manejo de excepciones
- Introducción a Git y GitHub:
  - Conceptos básicos (repositorio, commit, push, pull)
  - Crear y clonar repositorios
  - Comandos esenciales
- Ejercicios prácticos

Temario

CLASE 04 » Introducción a Pytest y Automatización Básica

- Introducción a Pytest:
  - Estructura de una prueba automatizada
  - Aserciones en Pytest
  - Fixtures básicas
- Automatización simple sin Selenium:
  - Testing de funciones Python
  - Verificaciones simples
- Ejercicios prácticos: Crear tests automatizados simples en Pytest

CLASE 05 » Introducción a HTML y Estructura de Páginas Web

- Estructura básica de HTML
- Elementos HTML comunes (divs, forms, inputs, buttons, etc.)
- Atributos HTML (id, class, name, value)
- Herramientas de desarrollo del navegador (DevTools)
- Inspección de elementos
- Ejercicios prácticos: Inspeccionar elementos en sitios web

CLASE 06 » DOM y CSS para Automatización

- Qué es el DOM (Document Object Model)
- Navegación por el DOM
- Selectores CSS básicos
- XPath: qué es y cómo utilizarlo
- Estrategias para localizar elementos en una página
- Ejercicios prácticos: Crear selectores para elementos
- Cuestionario

Temario

CLASE 07 » Introducción a Selenium WebDriver

- Qué es Selenium WebDriver
- Instalación paso a paso con ejemplos prácticos:
  - Instalación de Selenium
  - Configuración de WebDrivers (Chrome, Firefox)
  - Solución de problemas comunes
- Primeros scripts:
  - Abrir navegador
  - Navegar a una URL
  - Capturar título de página
  - Cerrar navegador
- Ejercicios prácticos: Configurar y ejecutar scripts básicos

CLASE 08 » Localización de Elementos y Acciones en Selenium

- Estrategias de localización en Selenium (ID, Name, XPath, CSS)
- Interacciones básicas (click, sendKeys, etc.)
- Manejo de formularios
- Esperas implícitas y explícitas
- Ejercicios prácticos
- Explicación del proyecto pre-entrega y requisitos

CLASE 09 » Page Object Model

- Introducción al patrón Page Object Model:
  - Concepto y beneficios
  - Estructura básica
  - Ventajas y mejores prácticas
- Implementación paso a paso:
  - Creación de clases para páginas
  - Métodos para interacción con elementos
  - Separación de pruebas y lógica de página
- Ejercicios prácticos: Refactorizar pruebas usando POM
- Revisión de avances del proyecto pre-entrega

Temario

CLASE 10 » Manejo de Datos de Prueba

- Parametrización de pruebas
- Lectura de datos desde archivos (CSV, JSON)
- Data-driven testing
- Generación de datos aleatorios para pruebas
- Ejercicios prácticos: Implementar tests parametrizados
- Cuestionario

CLASE 11 » Testing QA - Pruebas de API 2

- Introducción a REST API testing
- Uso de la biblioteca Requests
- Métodos GET y POST:
  - Obtener recursos
  - Crear recursos
  - Validación de respuestas
- Ejercicios prácticos: Probar APIs públicas con GET y POST

CLASE 12 » Automatización de Pruebas de API - Parte 2

- Métodos PUT, PATCH y DELETE:
  - Actualizar recursos
  - Eliminar recursos
- Validación avanzada de respuestas:
  - Códigos de estado
  - Contenido y estructura JSON
  - Headers y tiempos de respuesta
- Integración con Pytest
- Ejercicios prácticos y cuestionario

Temario

CLASE 13 » Reportes y Logging

- Generación de reportes de pruebas
- HTML Reports para Pytest
- Implementación de logging
- Capturas de pantalla automáticas
- Estrategias de gestión de errores
- Ejercicios prácticos: Configurar sistema de reportes completo

CLASE 14 » Introducción a BDD y Behave

- Fundamentos de Behavior-Driven Development
- Sintaxis Gherkin:
  - Feature
  - Scenario
  - Given-When-Then
- Instalación y configuración de Behave
- Ejemplos simples de features
- Ejercicios prácticos: Crear especificaciones en Gherkin
- Consignas del proyecto final

CLASE 15 » Integración Continua (CI/CD)

- Conceptos de CI/CD
- GitHub Actions para automatización:
  - Configuración paso a paso
  - Archivo YAML de workflow
- Ejemplo completo de pipeline:
  - Instalar dependencias
  - Ejecutar pruebas
  - Generar reportes
- Tiempo para proyecto final
- Resolución de dudas

Temario

CLASE 16 ❯❯ **Presentación del Proyecto Final y Cierre**

- Presentación de proyectos
- Feedback y evaluación
- Próximos pasos en la carrera de automation testing
- Recursos adicionales
- Cierre del curso

Proyecto Final Integrador:

Framework de Automatización de Pruebas

Los estudiantes desarrollarán un framework de automatización que combine pruebas de UI y API, aplicando buenas prácticas de diseño y generando reportes visuales.

Componentes requeridos:

- *Pruebas de UI (Selenium):*
  - Al menos 5 casos de prueba para un sitio web demo
  - Implementación del patrón Page Object Model
  - Manejo de diferentes escenarios (happy path y al menos un caso negativo)
  - Capturas de pantalla en caso de fallos
- *Pruebas de API:*
  - Al menos 3 casos de prueba usando requests
  - Cobertura de diferentes métodos HTTP (GET, POST, DELETE)
  - Verificación de respuestas y códigos de estado
- *Estructura y Características:*
  - Organización clara del código
  - Archivo README con instrucciones
  - Generación de reportes básicos

Proyecto Final Integrador:

Entregables

- Repositorio de GitHub con el código completo
- Presentación breve (5-10 minutos) demostrando las pruebas

Recursos Necesarios

- Computadora con conexión a Internet
- Cámara y micrófono
- Espacio en memoria para instalación de herramientas (ej. Visual Studio Code)

Requisitos para la acreditación

- 70% de asistencia a clases semanales de dos horas.
- Entrega de todos los desafíos obligatorios presentes a lo largo de la carrera.
- Aprobación del proyecto final integrador.

Bibliografía de Referencia

Selenium con Python (Documentación oficial)
La documentación oficial de Selenium para Python, que cubre todos los aspectos de la herramienta desde lo básico hasta funcionalidades avanzadas.
Enlace: **https://selenium-python.readthedocs.io/**

Pytest (Documentación oficial)
Documentación completa de Pytest, el framework de testing utilizado durante el curso, con explicaciones detalladas sobre aserciones, fixtures y plugins.
Enlace: **https://docs.pytest.org/**

Behave (Documentación oficial)
Guía oficial de Behave, la herramienta para implementar pruebas BDD en Python, con ejemplos y mejores prácticas.
Enlace: **https://behave.readthedocs.io/**

Python.org (Documentación oficial)
La documentación oficial de Python, recurso fundamental para aprender sobre todas las características del lenguaje.
Enlace: **https://docs.python.org/es/3/**


Buenos Aires
aprende
Agencia de Habilidades para el Futuro
Buenos
Aires
Ciudad